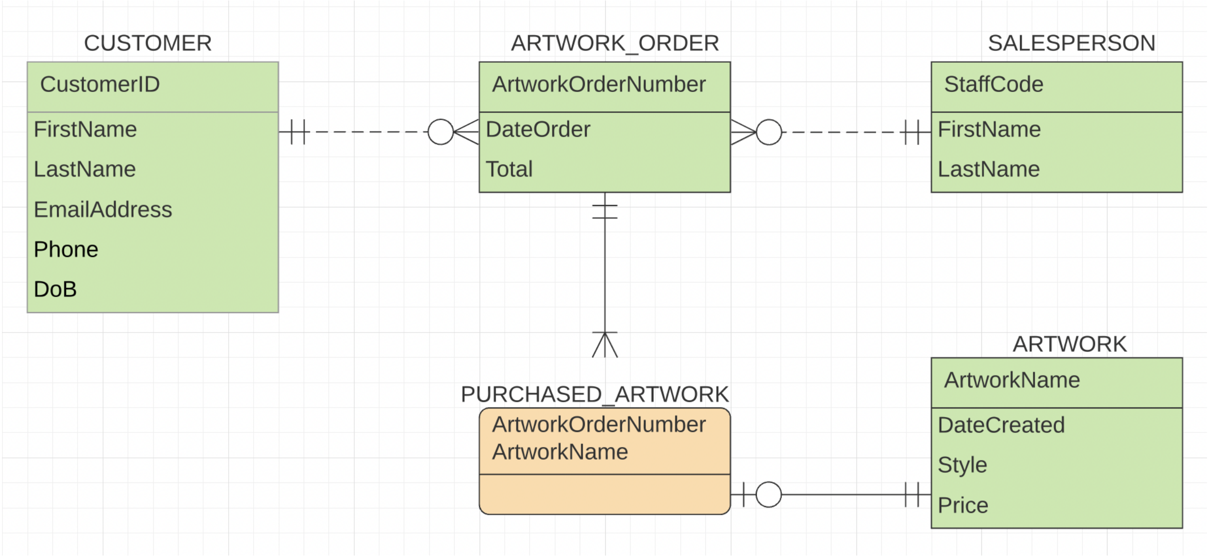
Name: Phan Thi Nguyet Que - 200168

Database: BrotherGallery

Description

Brother Gallery displays many artworks of famous artists around the world. Its collections include thousands of artworks with various topics as Literature and fantasy, war and conflict, Work of the 19th century still life, etc. Though the Gallery was just established a few months ago, it attracted a huge number of customers, who came to visit and buy artworks. The Gallery now wants a database to manage the Customer, Artwork, Order, and its Staff.

The entity relationship model diagram



Schemas of normalized tables

CUSTOMER(CustomerID, FirstName, LastName, EmailAddress, Phone, DoB)
ARTWORK(ArtworkNumber, ArtworkName, DateCreated, Style, Price)
SALESPERSON(StaffCode, FirstName,)
ARTWORK_ORDER(ArtworkOrderNumber, DateOrder, Total, *CustomerID*, *StaffCode*)
PURCHASED_ARTWORK(*ArtworkOrderNumber, ArtworkNumber*)

Relationship		Cardinality		
Parent	Child	Type	Max	Min
CUSTOMER	ARTWORK_ORDER	Non-identifying	1:N	M:O
SALESPERSON	ARTWORK_ORDER	Non-identifying	1:N	M:O
ARTWORK_ORDER	PURCHASED_ARTWORK	Identifying	1:N	M:M
ARTWORK	PURCHASED_ARTWORK	Identifying	1:1	M:O

Demonstration of the normalization

CUSTOMER.

1. Assumption**:**
    a. Every customer has just one email, phone.
    b. Customers are friends or family can take same email, phone
    c. Customers might have same DoB, same name
2. Functional dependencies:
    - CustomerID → (FirstName, LastName, EmailAddress, Phone, DoB)
3. Normalization: This table is a relation (NF1), in NF2 (no composite key), in NF 3 (no transitive dependencies), BCNF (whole key) and 4NF (no multivalued dependencies)

SALESPERSON.

1. Assumption:
2. Functional dependencies:
    - StaffCode → (FirstName, LastName)
    - (FirstName, LastName) → StaffCode
3. Normalization: This table is a relation (NF1), in NF2 (no composite key), in NF 3 (no transitive dependencies), BCNF (whole key) and 4NF (no multivalued dependencies)

ARTWORK.

1. Assumption:
    a. There are artwork with same name/datecreated/style/price
2. Functional dependencies:
    - ArtworkNumber → (ArtworkName, DateCreated, Style, Price)
3. Normalization: This table is a relation (NF1), in NF2 (no composite key), in NF 3 (no transitive dependencies), BCNF (whole key) and 4NF (no multivalued dependencies)

ARTWORK_ORDER.

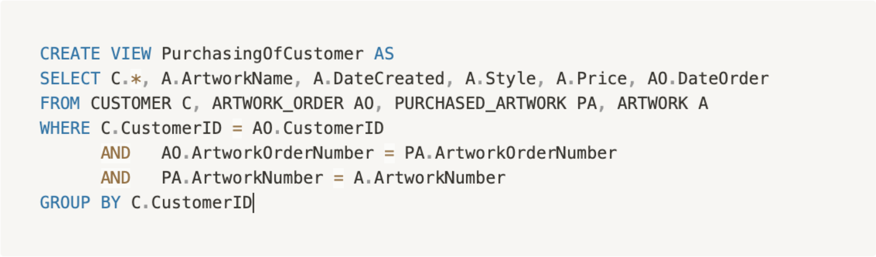
1. Assumption:
    a. Each date has many orders
    b. Each customer can has many order
    c. Each staff might relate to many order
2. Functional dependencies:
    - ArtworkOrderNumber → (DateOrder, Total, CustomerID, StaffCode)
3. Normalization: This table is a relation (NF1), in NF2 (no composite key), in NF 3 (no transitive dependencies), BCNF (whole key) and 4NF (no multivalued dependencies)

PURCHASED_ARTWORK.

All attributes are in Composite Primary Key → PURCHASED_ARTWORK is in 4NF

Answers in C level

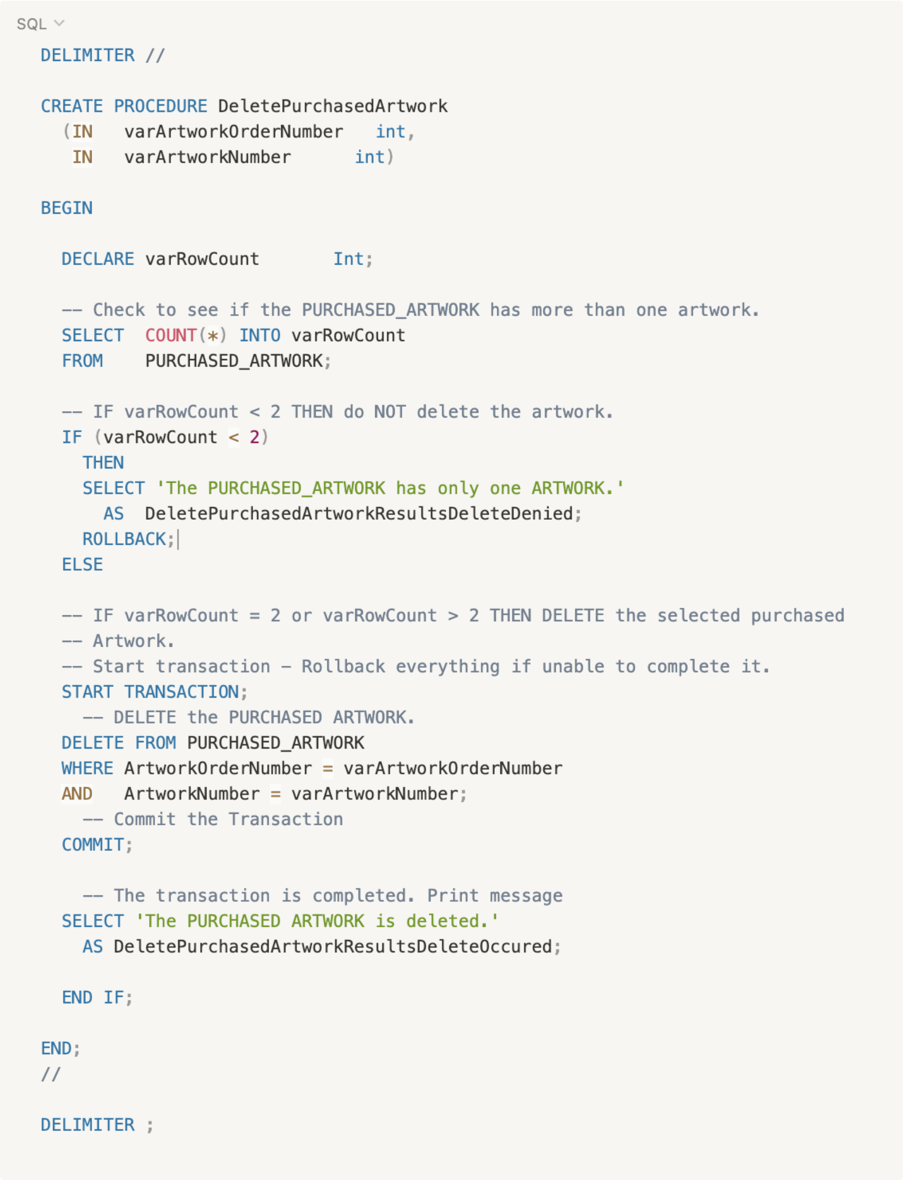
1. Write a view in your database that makes it easy to access some particularly useful collection of data or particularly painful sql code, e.g. the JOIN in B-1 below. Query that view



```sql
CREATE VIEW PurchasingOfCustomer AS
SELECT C.*, A.ArtworkName, A.DateCreated, A.Style, A.Price, AO.DateOrder
FROM CUSTOMER C, ARTWORK_ORDER AO, PURCHASED_ARTWORK PA, ARTWORK A
WHERE C.CustomerID = AO.CustomerID
      AND   AO.ArtworkOrderNumber = PA.ArtworkOrderNumber
      AND   PA.ArtworkNumber = A.ArtworkNumber
GROUP BY C.CustomerID
```

This PurchasingOfCustomer View contains information of each order: customer, artwork's detail info, order's info.

2. Write the procedure to prevent the last (mandatory) child from being deleted.



```sql
DELIMITER //

CREATE PROCEDURE DeletePurchasedArtwork
  (IN   varArtworkOrderNumber   int,
   IN   varArtworkNumber      int)

BEGIN

  DECLARE varRowCount       Int;

  -- Check to see if the PURCHASED_ARTWORK has more than one artwork.
  SELECT  COUNT(*) INTO varRowCount
  FROM    PURCHASED_ARTWORK;

  -- IF varRowCount < 2 THEN do NOT delete the artwork.
  IF (varRowCount < 2)
    THEN
    SELECT 'The PURCHASED_ARTWORK has only one ARTWORK.'
      AS  DeletePurchasedArtworkResultsDeleteDenied;
    ROLLBACK;
  ELSE

  -- IF varRowCount = 2 or varRowCount > 2 THEN DELETE the selected purchased
  -- Artwork.
  -- Start transaction - Rollback everything if unable to complete it.
  START TRANSACTION;
    -- DELETE the PURCHASED ARTWORK.
  DELETE FROM PURCHASED_ARTWORK
  WHERE ArtworkOrderNumber = varArtworkOrderNumber
  AND   ArtworkNumber = varArtworkNumber;
    -- Commit the Transaction
  COMMIT;

    -- The transaction is completed. Print message
  SELECT 'The PURCHASED ARTWORK is deleted.'
    AS DeletePurchasedArtworkResultsDeleteOccured;

  END IF;

END;
//

DELIMITER ;
```

The above procedure aims at preventing deleting the last row in the PURCHASED_ORDER table. To check this procedure:

Pass in the SQL code of Procedure in the console, and CALL the procedure with the following guide.

(1) To delete a purchased_order
CALL DeletePurchasedArtwork (10001, 4);
CALL DeletePurchasedArtwork (10002, 1);
CALL DeletePurchasedArtwork (10003, 3);

(2) To generate error message:
CALL DeletePurchasedArtwork (10004, 2);

Answers in B level

Prompt	Description	Input	URL
1. Query your database for information from at least two tables.	This interface will return the total sales of the selected staff at Brother Gallery.	StaffCode: - BGS01 - BGS02	http://103.42.58.103/plesk-site-preview/s200168.fuv.edu.vn/https/103.42.58.103/QueryStaffSales.php
2. Insert a row into the parent table.	This interface will inserted new artwork and its information into the database (All inputs are required)	- ArtworkName - DateCreated:YYYY-MM-DD - Style - Price	http://103.42.58.103/plesk-site-preview/s200168.fuv.edu.vn/https/103.42.58.103/insertNewArtwork.php
3. Update a row in that parent table	This interface will update Artwork information by ArtworkNumber. - With the field that doesn't need to be updated, please enter "NA". - With fields need to be updated, please specified new data	- ArtworkNumber: current ArtworkNumber - ArtworkName: New Data/"NA" - DateCreated: New data/ "NA" - Style: New data/"NA" - Price: New data/"NA"	http://103.42.58.103/plesk-site-preview/s200168.fuv.edu.vn/https/103.42.58.103/UpdateArtwork.php

Answers in A level

Prompt	Description	Input	URL
1. Query your database for information from three tables (include an intersection table) using a JOIN	This interface will return information of each order includes data from 3 tables: - ARTWORK_ORDER - ARTWORK - PURCHASED_ARTWORK	ArtworkOrderNumber Note: After such changes in the above part, the current PURCHASED_ARTWORK just has one row with ArtworkOrderNumber 10004.	http://103.42.58.103/plesk-site-preview/s200168.fuv.edu.vn/https/103.42.58.103/Query3Tables.php
2. Insert a row into	This interface will insert data	- ArtworkOrderNumber of	http://103.42.58.103/ple

one of the N:M intersection tables in your database	into PURCHASED_ARTWORK	Artwork_Order that is not in PURCHASED_ARTWORK. - ArtworkNumber of Artwork that is not in PURCHASED_ARTWORK	sk-site-preview/s200168.fuv.edu.vn/https/103.42.58.103/Insert_NM_tables.php